

**Application No.: 10/715,121**

**REMARKS**

Claims 1-10 were pending at the time of the Action. Claims 1, 6, 7, 8, and 9 are amended. Claims 1-10 are currently active for examination. Claims 1, 6, 7, 8, and 9 are the only independent claims.

**Claims 1, 6, and 9 were objected to regarding alleged informalities.** Applicants submit that this objection have been overcome by the foregoing amendments.

**Claims 1, 3, 5, and 7-10 were rejected under 35 U.S.C. 112, first paragraph,** as allegedly failing to comply with the written description requirement. The claims contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventors, at the time the application was filed. Applicants traverse this rejection.

Specifically, the Office Action, states that in claims 1, 3, 5 and 7-9, the recitation of "detect/detecting/detection/detected a change of part of a WSDL file defining" and "wherein each user process in the second set of user processes has the second zone identifier associated therewith" are allegedly not disclosed in the specification. Applicants disagree.

First, Applicants submit that the term "wherein each user process in the second set of user processes has the second zone identifier associated therewith" does not occur in the claims, and therefore the rejection is improper with respect to this term.

Second, Applicants submit that the recitations regarding detecting a change of a WSDL file are described, at a minimum, by the specification at page 2, lines 4-6, which state, "The WSDL (Web Service Description Language) is a format of the structured document for defining the interface of the Web Service," and by the specification at page 9, line 26 to page 11, line 12, which states:

**Application No.: 10/715,121****2. Outline of Automatic Change and Replacement of Client Program**

Description is now made to the summarized flow of revision of the interface definition information by the Web service server 3 and automatic change and replacement of the client program by the client program server 1.

- (1) In the Web service server 3, the **interface definition information** is revised and the server program is changed in accordance with a new interface.
- (2) The Web service server 3 re-registers a storage location of the revised interface definition information in the UDDI registry 4.
- (3) The user computer 2 utilizes the Web service through the client program server 1.
- (4) The client program in the client program server 1 transmits a request message to the Web service server 3. However, **the client program is prepared in accordance with the unrevised interface definition information** and accordingly a new server program is not fit to the unrevised interface definition information, so that **an error is returned to the client program**.
- (5) The client program server 1, when it is confirmed that the error indicates mismatch of the interface definition information, stops the client program temporarily. On the other hand, the client program server 1 causes the user computer 2 to display the message such as "Error is detected. Wait a moment".
- (6) The client program server 1 accesses to the UDDI registry 4 to obtain the revised interface definition information of the Web service server 3 and compares it with the unrevised interface definition information.
- (7) The client program server 1 **extracts portions related to the change of the interface** from source codes of modules constituting the client program and changes the portions to be fit to the revised interface definition information.
- (8) The client program server 1 converts the source codes of the modules constituting the client program into an executable form to **replace the current client program** thereby.
- (9) The client program server 1 causes the user computer 2 to display the message so that input for utilization of the Web service is made again.
- (10) The replaced client program transmits the request message to the Web service server 3 in response to a request from the user.

Thus, Applicants submit that claims 1, 3, 5, and 7-10 satisfy the written description requirement of 35 U.S.C. 112, first paragraph.

**Claims 1-6 were rejected under 35 USC 112, second paragraph**, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. Specifically, the Office Action states that in claims 1 and 6 the term "said changed client program" in lines 13 and 20 respectively has insufficient antecedent basis.

Applicants submit that this rejection has been overcome by the foregoing amendments.

**Application No.: 10/715,121**

**Claims 7 and 8 were rejected under 35 USC 101** because the claimed invention is allegedly directed to non-statutory subject matter. Specifically, the Office Action asserts that the claim 7 term "client program management apparatus" and the claim 8 term "client program management group" are software per se.

Applicants submit that this rejection has been overcome by the foregoing amendments which utilize the term "computer program storage media."

**Claims 1-2 and 6-10 are rejected under 35 USC 103(a)** as being unpatentable over US Patent Application Publication No. 2003/0220925 A1 to Lior in view of US Patent Application Publication 2004/0133656 A1 to Butterworth et al. Applicants traverse this rejection.

**Claims 3-5 are rejected under 35 USC 103(a)** as being unpatentable over US Patent Application Publication 2003/0220925 A1 to Lior in view of US Patent Application Publication 2004/0133656 A1 to Butterworth et al. as applied to claim 1, and further in view of US Patent Application Publication 2004/0117199 A1 to Fremantle et al. Applicants traverse this rejection.

Independent claim 1 recites, in part:

transmitting an access request from the client program to a server program;  
when an error occurs, analyzing a cause of the error;  
**detecting a change part of a WSDL file defining an interface of said server program when the error is analyzed to be caused by change of the WSDL file defining said interface of said server program in the analyzing step;**  
**deciding a change portion and a changing scheme of said client program related to said change of said WSDL file defining said interface to change said client program;**  
**dynamically replacing the client program being currently operated and which transmitted the request in which the error occurred with a changed client program; and**  
re-transmitting the access request to the server program from the said changed client program.

**Application No.: 10/715,121**

In other words, the server program and the client program work together as a pair, so that when the server program interface is changed, then the corresponding client program portion should be changed. In claim 1, the change is detected when the cause of an error is analyzed, then the corresponding client program portion is identified, then the client program is **dynamically** replaced. Note that the client computer is automatically updated without stopping the client computer (the client program is dynamically replaced).

In contrast to claim 1, Lior, at paragraphs [0010], [0011], [0012], [0041], [0056], and [0057] states merely:

[0010] Another aspect of the invention is to **maintain the provisioned information by the periodic processing of the WSDL document.**

[0011] In one aspect of the present invention, there is provided a method of provisioning Web services. The method comprises the steps of receiving a Web services description language document of a Web service, automatically extracting information associated with the Web service, and **automatically generate a record** of the information for use in access policies of the Web service.

[0012] In another aspect of the present invention, there is provided a method for updating a Web services provisioning information file. The method comprises the steps of **receiving notice of a change** to a Web service, automatically obtaining a Web services description language document of the Web service, automatically extracting changed information of the Web services description language document, and automatically updating a recorded entry of the Web service in the Web services provisioning information file.

[0041] the URL used by an application 11 to access the temperature Web service;

[0056] The WSDL document monitoring unit, on a periodically configurable interval, may check the WSDL documents that the unit has been assigned to manage...

[0057] By **having the access control provisioning module 21 periodically monitor any changes to the WSDL document files**, alerts may be generated when human intervention will be required, for example, when changes in the Web services require changes in the access policies or fault detection policies. Furthermore, by monitoring the well known locations where WSDL documents are published, such as a directory in a file system or UDDI registry, the access control provisioning module 21 can detect the presence of new Web services...

**Application No.: 10/715,121**

Thus, Lior merely publishes the WSDL document to facilitate periodic processing ("periodically monitor any changes") of the WSDL document at paragraph [0057]. Paragraphs [0040] and [0041] show merely the information items to be extracted from the published WSDL document. Paragraphs [0062] and [0063] discuss merely maintaining the WSDL information.

In contrast to Lior, claim 1 requires, **"detecting a change part of a WSDL file defining an interface of said server program when the error is analyzed to be caused by change of the WSDL file."** Also in contrast to Lior, claim 1 also requires, **"dynamically replacing the client program being currently operated and which transmitted the request in which the error occurred."**

Butterworth, at paragraphs [0067], [0093], and [0094], states merely:

[0067] Web services sometimes encounter **errors during message processing**. These errors can occur when the client formats the message incorrectly, when the web service does not understand the message, or when the web service produces an exception. The fault phase 910 handles such errors. For example, the fault phase defines a special sequence of actions to be performed when the agent receives a SOAP fault from the web service. For example, **request and fault messages may be written to a fault log that can be examined by the system administrator.**

[0093] If the response is a fault (block 1035--"Yes"), then the fault processing of FIG. 10C is invoked (block 1037). If the response is not a fault (block 1035--"No"), then response processing proceeds in a manner similar to request processing. The main difference is that response processing uses the same context as the request processing instead of creating a new context. When response processing is initiated, the agent changes the operation name to match the response. The agent then finds the action list for the name and begins the action processing. Thus, output phase processing operations 1036-1046 correspond to previously described input phase processing operations 1017-1027. The final processing operation of FIG. 10B is to return the responses to the client (block 1048).

[0094] Fault processing is illustrated in FIG. 10C. Fault processing is performed in the same manner as response processing. Fault processing is invoked when the system detects a fault indication in the response message, as shown with block 1037 of FIGS. 10B and 10C. Processing blocks 1050-1062 of

**Application No.: 10/715,121**

FIG. 10C correspond to previously described processing operations 1017-1027 of FIG. 10A.

Thus, Butterworth does not teach or suggest, “detecting a change part of a WSDL file defining an interface of said server program when the error is analyzed to be caused by change of the WSDL file,” and does not teach or suggest “dynamically replacing the client program being currently operated and which transmitted the request in which the error occurred.”

Thus, at a minimum, the combination of Lior and Butterworth fails to teach or suggest either of the forgoing elements of claim 1 (detecting the change when the error is analyzed, and dynamically replacing the client program), and therefore claim 1 is allowable over the cited art.

Independent method claim 6 is similar to method claim 1. Independent apparatus claim 7, independent computer program storage media claim 8, and client/server system claim 9 are analogous to method claim 1. Thus, Applicants submit that independent claims 6-9 are allowable for reasons similar to claim 1.

Under Federal Circuit guidelines, a dependent claim is allowable if the independent claim upon which it depends is allowable because all the limitations of the independent claim are contained in the dependent claims, *Hartness International Inc. v. Simplimatic Engineering Co.*, 819 F.2d at 1100, 1108 (Fed. Cir. 1987).

Thus, as independent claims 1 and 9 are allowable for the reasons set forth above, it is respectfully submitted that dependent claims 2-5 and 10 are allowable for at least the same reasons as their respective base claims.

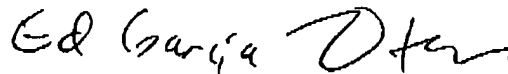
**Application No.: 10/715,121**

Accordingly, it is urged that the application, as now amended, is in condition for allowance, an indication of which is respectfully solicited. If there are any outstanding issues that might be resolved by an interview or an Examiner's amendment, Examiner is requested to call the undersigned attorney at the telephone number shown below.

To the extent necessary, a petition for an extension of time under 37 C.F.R. 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 500417 and please credit any excess fees to such deposit account.

Respectfully submitted,

McDERMOTT WILL & EMERY LLP



Eduardo Garcia-Otero  
Registration No. 56,609

600 13<sup>th</sup> Street, N.W.  
Washington, DC 20005-3096  
Phone: 202.756.8000 KEG/EG:mjb  
Facsimile: 202.756.8087  
**Date: May 22, 2008**

**Please recognize our Customer No. 20277  
as our correspondence address.**

WDC99 1571096-1.062807.0150